НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ ЗАХИСТУ ІНФОРМАЦІЇ

«До захисту допущено»
В.о. завідувача кафедрою
_____ М.М.Савчук
(підпис) (ініціали, прізвище)
“___”_____20 _ р.

Дипломна робота

на здобуття ступеня бакалавра

з напряму підготовки : 113 «Прикладна математика»
(код і назва)

на тему: Порівняльний аналіз схем автентифікованого шифрування на основі налаштованого блокового шифру (на прикладі учасників конкурсу CEASAR). _
_____
_____

Виконав (-ла): студент (-ка) 4 курсу, групи ФІ-62
(шифр групи)

_____ Олефір Поліна Юріївна _____ _____
(прізвище, ім’я, по батькові) (підпис)

Керівник доцент кафедри ММЗІ, Завадська Л. О. _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент_____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших авторів
без відповідних посилань.
Студент _____
(підпис)

Київ – 2020року

Національний технічний університет України
«Київський політехнічний інститут
імені Ігоря Сікорського»
Фізико-технічний інститут

Кафедра математичних методів захисту інформації

Рівень вищої освіти – перший (бакалаврський)

Напрям підготовки - 113 «Прикладна математика»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедрою

М.М.Савчук

_____
(підпис) (ініціали, прізвище)

«___»_____20__ р.

ЗАВДАННЯ
на дипломну роботу студенту

Олефір Поліна Юріївна
(прізвище, ім'я, по батькові)

1. Тема роботи порівняльний аналіз схем автентифікованого шифрування на основі налаштованого блокового шифру (на прикладі учасників конкурсу CEASAR),
керівник роботи доцент кафедри ММЗІ, Завадська Людмила Олексіївна,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від _____ р. № _____

2. Термін подання студентом роботи _____

3. Вихідні дані до роботи _____
_____
_____
_____

4. Зміст роботи _____
_____
_____
_____

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо) _____

_____

_____

_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Провести огляд опублікованих джерел за тематикою дослідження	25.10	
2.	Дослідити, яким чином застосовується метод інтегрального криптоаналіу до налаштованих блокових шифрів типу AES і до схем автентифікованого шифрування побудованих на них	11.03	
3.	Виявити вади тих алгоритмів, які дозволили застосувати до них криптоатаки, і властивості переможців конкурсу, які забезпечують їх стійкість	23.04	
4.	Зробити висновки з проведеного аналізу	26.05	

Студент _____ Олефір П.Ю.
(підпис) (ініціали, прізвище)

Керівник роботи _____ Завадська Л. О.
(підпис) (ініціали, прізвище)

РЕФЕРАТ

Кваліфікаційна робота містить: 50 стор., 13 рисунків, 3 таблиці, 15 джерел.

Метою роботи є порівняльний аналіз схем автентифікованого шифрування з асоційованими даними на основі налаштованих блокових шифрів, які представлені у конкурсі CAESAR (Silver, Kiasu, OCB).

Об'єктом дослідження є інформаційні процеси в системах криптографічного захисту.

Предметом дослідження є криптографічні алгоритми автентифікованого шифрування на основі налаштованих блокових шифрів.

У роботі зроблено порівняльний аналіз деяких схем автентифікованого шифрування на основі налаштованих блокових шифрів, які представлені у конкурсі CAESAR. Проаналізовано особливості будови, способи налаштування (уведення 'твіку') та криптоатаки, що були застосовані до алгоритмів Silver, Kiasu. Також розглянутий метод інтегрального криптоаналізу як теоретична основа для згаданих атак.

АВТЕНТИФІКОВАНЕ ШИФРУВАННЯ, НАЛАШТОВАНИЙ (TWEAKABLE) БЛОКОВИЙ ШИФР, КОНКУРС CAESAR, ІНТЕГРАЛЬНИЙ КРИПТОАНАЛІЗ

ABSTRACT

The diploma work contains: 50 pages, 13 drawing, 3 tables, 15 sources

The aim of the work is a comparative analysis of the authenticated encryption schemes with associated data based on tweakable block ciphers presented in the CEASAR competition (Silver, Kiasu, OCB).

The object of the research are information processes in cryptographic security systems.

The subject of the research are cryptographic algorithms of the authenticated encryption based on tweaked block ciphers.

A comparative analysis of the authenticated schemes based on the tweakable block ciphers presented in the CAESAR competition is performed. The structure features, the methods of configuration (introduction of 'tweak') and cryptographic attacks that were applied to the algorithms of Silver, Kiasu are analyzed. The method of integral cryptanalysis as a theoretical basis for the mentioned attacks is also considered.

AUTHENTICATED ENCRYPTION, CAESAR COMPETITION, TWEAKABLE BLOCK CIPHERS, CAESAR COMPETITION, INTEGRAL CRYPTANALYSIS

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

$\oplus$ — операція побітового додавання

AE — автентифіковане шифрування

AEAD — автентифіковане шифрування з асоційованими даними

MAC — код аутентифікції повідомлення

AD — асоційовані дані

ВТ — відкритий текст

ШТ — шифрований текст

ВСТУП

Актуальність дослідження. На даний час важливим аспектом людської діяльності є конфіденційність даних, фінансових транзакцій, бізнес-операцій та безпечне онлайн-спілкування. З кожним днем збільшується кількість інтернет-користувачів. З розвитком комп'ютерних технологій зростає обчислювальна потужність пристроїв – це призводить до того, що зловмисники мають більш високі можливості перехоплення повідомлень. Існують випадки, коли надіслане зашифроване повідомлення може перехопити, підробити противник та надіслати його модифікацію отримувачу. Звичайні симетричні алгоритми шифрування забезпечують в основному конфіденційність повідомлення. Тому наразі важливо забезпечити не тільки конфіденційність, а й цілісність та автентичність повідомлення, які гарантують цілісність інформації. Саме з цією метою були створені симетричні криптографічні алгоритми автентифікованого шифрування, які одночасно дозволяють забезпечити як конфіденційність, так і автентичність інформації. Такі схеми мають переваги у порівнянні з окремими застосуваннями алгоритмів шифрування та автентифікації.

У 2014 році розпочався всесвітній конкурс під назвою CAESAR. Ця абревіатура означає «Competition for Authenticated Encryption: Security, Applicability and Robustness» , тобто «Конкурс Аутентифікованого Шифрування: Безпека, Застосовність, Надійність». Змагання було започатковане за ініціативи Національного інституту стандартів та технологій США (NIST). Метою даного конкурсу є виявлення автентифікованих схем шифрування, які у своїй програмній реалізації одночасно можуть досягнути безпеки, ефективності та надійності. Одним із найпоширеніших способів підвищення їх ефективності та стійкості є побудова налаштованих (tweakable) систем автентифікованого шифрування. Велика кількість учасників конкурсу CAESAR є саме

такими схемами, що говорить про перспективність даного напрямку. Слід зазначити, що не всі налаштовані алгоритми автентифікованого шифрування виявилися стійкими. Багато в чому їх стійкість залежить від способу налаштування (уведення твіку), який постійно вдосконалюється. Аналіз будови налаштованих систем автентифікованого шифрування, їх стійкості та ефективності є актуальним і корисним для розвитку даної галузі криптографії.

Метою дослідження є порівняльний аналіз схем автентифікованого шифрування з асоційованими даними на основі налаштованих блокових шифрів, які представлені у конкурсі CAESAR (Silver, Kiasu, OCB).

Задачею дослідження є вивчення впливу способів налаштування блокових шифрів на стійкість схем автентифікованого шифрування з асоційованими даними на прикладі алгоритмів Silver, Kiasu та OCB. Дослідження теоретичних основ застосування методу інтегрального криптоаналізу до схем такого типу, а також виявлення недоліків схем, які дозволяють розробити атаки на них з використанням інтегрального аналізу.

Для досягнення сформованої мети необхідно виконати наступні **завдання дослідження**, які були виконані у роботі:

1) провести огляд опублікованих джерел за тематикою дослідження;

2) проаналізувати учасників конкурсу CAESAR з точки зору їх будови, виявити ті, які побудовані на налаштованих блокових шифрах;

3) дослідити, яким чином застосовується метод інтегрального криптоаналіу до налаштованих блокових шифрів типу AES і до схем автентифікованого шифрування побудованих на них;

4) виявити вади тих алгоритмів, які дозволили застосувати до них криптоатаки, і властивості переможців конкурсу, які забезпечують їх стійкість;

5) зробити висновки з проведеного аналізу.

Об'єктом дослідження є інформаційні процеси в системах

криптографічного захисту.

Предметом дослідження є криптографічні алгоритми автентифікованого шифрування на основі налаштованих блокових шифрів.

При розв'язанні поставлених завдань використовувались такі *методи дослідження*: методи криптографічного аналізу, теорія складності алгоритмів та теорія ймовірностей.

Наукова новизна отриманих результатів полягає у тому, що зроблено порівняння саме тих алгоритмів автентифікованого шифрування, які побудовані на налаштованих блокових шифрах і до яких був застосований метод інтегрального криптоаналізу. Виявлені недоліки тих алгоритмів, які були відсіяні протягом конкурсу і переваги переможців.

Практичне значення результатів полягає у тому, що вони висвітлюють причини різниці у стійкості схем автентифікованого шифрування, побудованих на налаштованих блокових шифрах і, таким чином, можуть допомогти у виборі найбільш стійких і ефективних з них при побудові систем захисту інформації.

1 ОСНОВНІ ПОНЯТТЯ ТА ОЗНАЧЕННЯ З ТЕОРІЇ АВТЕНТИЧНОГО ШИФРУВАННЯ. ОГЛЯД КОНКУРСУ CAESAR

У даному розділі наведено основні поняття та означення теорії автентифікованого шифрування. Розглянуто конкурс CAESAR, наведені вимоги до учасників, їх класифікація.

1.1 Необхідні теоретичні відомості з теорії автентифікованого шифрування

Автентифікація та MAC. Використання додаткового алгоритму код автентифікації повідомлення (MAC) дозволяє перетворити схему шифрування в схему автентифікованого шифрування. MAC - це алгоритм, який за допомогою секретного ключа обробляє вхідні дані для автентифікації та повертає тег автентифікації фіксованої довжини (дивись рисунок 1.1). Часто коди автентифікації повідомлень представлені як геш-функції тому, що на вхід подаються дані будь-якого розміру, а на виході отримуємо фіксованого. Різницею між MAC та геш-функцією є те, що MAC потребує секретного ключа для перевірки обчислення MAC відправником. Для забезпечення безпеки використовуються різні ключі: один для шифрування, інший для MAC. При передачі повідомлення по незахищеному каналу користувачі A та B мають бути впевненими про незмінність повідомлення, тобто можливість перевірки правильності тегу. Користувач A додає до повідомлення тег, який отримує за допомогою алгоритму MAC та за допомогою спільного секретного ключа надсилає повідомлення користувачу B. Коли користувач B отримав повідомлення, він обчислює тег автентифікації, використовуючи той самий спосіб, та перевіряє рівність тегів. Якщо теги

рівні, то повідомлення цілісне.

Означення 1.1. Код автентифікації повідомлення $\sqcap$ складається з трьох алгоритмів: $\sqcap = <\mathcal{K}, \mathcal{G}, \mathcal{V}>$, де

1) $\mathcal{K}$ — алгоритм генерації ключів. Результатом алгоритму є множина ключів K.

2) $\mathcal{G}$ — алгоритм генерації тегів. Генерація тегів відбувається шляхом застосування функції MAC до ключа та повідомлення ($\tau \leftarrow MAC_K(M)$). Значення τ - це двійковий вектор фіксованої довжини.

3) $\mathcal{V}$ — алгоритм перевірки тегів приймає на вхід ключ, повідомлення та тег, отриманий в результаті функції перевірки тегів. У результаті отримуємо біти 1 або 0, які позначають успішність перевірки відповідно.

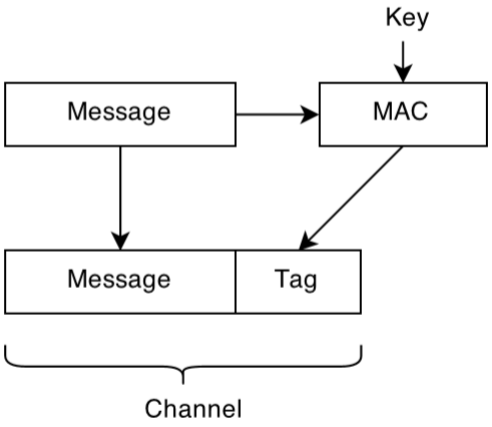

Рисунок 1.1 – Схема MAC-кода

Перші загальні схеми, що поєднують автентифікацію та шифрування були розглянуті у роботі [1], у якій і був уведений термін 'автентифіковане шифрування'. Усі три схеми (дивись рисунок 1.2,1.3,1.4) вважаються безпечними при використанні Nonce, тільки схема 'шифрування-потім-MAC' є безпечною, якщо Nonce не використовується.

Загальні схеми побудови аутентифікованих шифрів:

1) Шифрування — та — MAC: $AE_k(M) = E_k(M)||\tau_k(M)$

2) Шифрування — потім – MAC: $AE_k(M) = E_k(M)||\tau_k(E_k(M))$

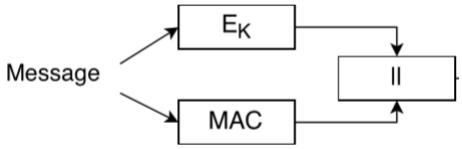

Рисунок 1.2 – Шифрування — та — MAC

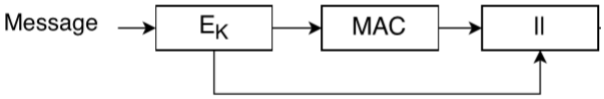

Рисунок 1.3 – Шифрування — потім – MAC

3) MAC — потім — шифрування: $AE_k(M) = E_k(\tau_k(M)\|M)$

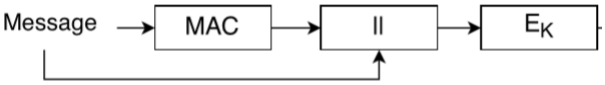

Рисунок 1.4 – MAC — потім — шифрування

Існують декілька способів реалізації коду автентифікації повідомлення, а саме на основі геш-функції (HMAC), на основі спеціальних режимів блокового шифру (CBC-MAC, PMAC, CMAC), на основі універсальних геш-функцій.

Автентифіковане шифрування (AE). Автентифіковане шифрування — симетрична схема шифрування, яка забезпечує як конфіденційність так і автентичність інформації.

Означення 1.2. Nonce — це унікальний номер, який не повторюється, використовується не більше одного разу, може бути не випадковим.

Означення 1.3. Автентифікований шифр — це шестірка виду:
$\Xi = < \mathcal{M}, \mathcal{K}, \mathcal{C}, \tau, \mathcal{N}, \mathcal{E}, \mathcal{D} >$, де
$\mathcal{M}$ — множина повідомлень;
$\mathcal{N}$ — множина ключів;

$\mathcal{C}$ — множина шифротекстів;

τ — тег;

$\mathcal{N} - nonce$;

$\mathcal{E} : \mathcal{M} \times \mathcal{N} \times \mathcal{K} \to \mathcal{C} \times \tau$ — шифруюче перетворення;

$\mathcal{D} : \mathcal{C} \times \tau \times \mathcal{N} \times \mathcal{K} \to \mathcal{M} \times \tau$ — розшифровуюче перетворення.

Алгоритми автентифікованого шифрування в залежності від обробки повідомлення поділяють на дві групи: перша називається однопрохідною, де повідомлення обробляється тільки один раз. Така схема називається "ОСВ". Друга — двопрохідна, де обробка повідомлення відбувається двічі.

Безпека алгоритмів AE. У роботі [1] Белларе та Нампремпре зазначено вимоги до безпеки схем автентифікованого шифрування, а саме IND-CPA, INT-CTXT. Позначимо $K \xleftarrow{\$} \mathcal{K}$ - будь-який ключ з множини ключів. Вираз $P(\mathcal{A}^{\mathcal{O}} \Rightarrow 1)$ означає ймовірність того, що противник виведе 1 після взаємодії з оракулом.

Схема автентифікованого шифрування безпечна, якщо отримані шифротексти не відрізняються від випадкових шифротекстів, які видав фейковий оракул та вона називається нерозрізненою схемою при вибраному відкритому тексті (IND-CPA).

Існують дві моделі криптоаналізу схем AE: **nonce-respecting** та **nonce-repeating**.

Означення 1.4. Нехай, $\mathcal{A}$ противник з доступом до шифруючого оракула $\mathcal{E}_k(\cdot, \cdot)$. Цей оракул приймає на вхід пару (N, M), де N - *nonce* та M - повідомлення, повертає шифротекст $C \leftarrow \mathcal{E}_k(N, M)$. Нехай, $(N_0, M_0), \ldots, (N_{s-1}, M_{s-1})$ - послідовність запитів до оракула. Противник $\mathcal{A}$ називається $nonce - respecting$, якщо $N_0, \ldots, N_{s-1}$ є завжди різними, незважаючи на відповідь оракула та монету противника $\mathcal{A}$.

Вважаємо, що противник $\mathcal{A}$ nonce-respecting з доступом до двох оракулів: перший оракул $\mathcal{R}(\cdot, \cdot)$ повертає випадкові біти, позначені як C довжина, яких дорівнює довжині відкритого тексту $C \leftarrow \{0,1\}^n$, де $n = |M|$, ці біти приймаються є фейковими. Другий оракул $\mathcal{E}_k(\cdot, \cdot)$

використовує алгоритм шифрування $C \leftarrow \mathcal{E}_k(N, M)$. Спочатку підкидається монета. У залежності від результату противник запитує або першого, або другого оракула. Мета противника — визначити результат підкидання монети шляхом випадкового припущення та відрізнити випадкові біти від дійсного шифрування.

Означення 1.5. Безпека IND-CPA. Нехай, $\Xi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ - автентифікована схема шифрування. Перевага IND-CPA над противником $\mathcal{A}$, який має доступ до шифруючого оракула або до $\mathcal{E}_k(\cdot,\cdot)$, або до $\mathcal{R}(\cdot,\cdot)$ визначена як:

$$Adv_{\Xi}^{IND-CPA}(\mathcal{A}) = |P(K \leftarrow^{\$} \mathcal{K} : \mathcal{A}^{\mathcal{E}_k(\cdot,\cdot)} \Rightarrow 1) - P(\mathcal{A}^{\mathcal{R}(\cdot,\cdot)} \Rightarrow 1)|$$

Автентифікація визначається поняттям цілісності шифрованих текстів (INT-CTXT). Цілісність шифротексту означає, що неможливо згенерувати дійсний шифрований текст, не створеним відправником. Нехай $\Xi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ - автентифікована схема шифрування. Маємо противника $\mathcal{A}$, який є nonce-respecting та має доступ до шифруючого оракула $\mathcal{E}_k(\cdot,\cdot)$ та оракула розшифрування $\mathcal{D}_k(\cdot,\cdot)$. Противник відправляє будь-яке повідомлення до оракула шифрування, та може отримати дійсний шифрований текст або знак ?. Мета противника — підробити шифротекст, який приймає оракул для розшифрування та повертає BT. Що стосується способу підробки, то противник може використовувати той же nonce, що і в попередньому запиті. Але противник не може зробити запит з шифрованим текстом, який отримав у попередньому запиті для розшифрування, бо в такому випадку результат буде успішним.

Означення 1.6. Безпека INT-CTXT. Нехай, $\Xi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ - автентифікована схема шифрування. Перевага INT-CTXT над противником $\mathcal{A}$, який має доступ до шифруючого оракула $\mathcal{E}_k(\cdot,\cdot)$ та до оракула розшифрування $\mathcal{D}(\cdot,\cdot)$ визначається як:

$$Adv_{\Xi}^{IND-CTXT}(\mathcal{A}) = |P(K \leftarrow^{\$} \mathcal{K} : \mathcal{A}^{\mathcal{E}_k(\cdot,\cdot),\mathcal{D}(\cdot,\cdot)} \Rightarrow forges)|$$

Нерозрізненість при обраній атаці на шифрований текст (CCA3) полягає у тому, що противник має вгадати до якого з вибраних відкритих текстів належить шифрований текст, який він отримав від оракула. Противник може додатково розшифровувати будь-який зашифрований текст, який отримав від оракула.

Означення 1.7. Безпека CCA3. Нехай, $\Xi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ - автентифікована схема шифрування. Перевага CCA3 над противником $\mathcal{A}$, який має доступ до шифруючого оракула $\mathcal{E}_k(\cdot,\cdot)$ та до оракула розшифрування $\mathcal{D}(\cdot,\cdot)$ визначається як:

$$Adv_{\Xi}^{CCA3}(\mathcal{A}) = |P(K \leftarrow^{\$} \mathcal{K} : \mathcal{A}^{\mathcal{E}_k(\cdot,\cdot),\mathcal{D}(\cdot,\cdot)} \Rightarrow 1) - P(K \leftarrow^{\$} \mathcal{K} : \mathcal{A}^{\mathcal{R}(\cdot,\cdot),\mathcal{D}(\cdot,\cdot)} \Rightarrow 1)|$$

Автентифікована схема шифрування, якщо досягає INT-CTXT та IND-CPA, то вона і досягає CCA3.

Більш строгим поняттям безпеки для схем AE є поняття неправомірного використання AE (MRAE). Це поняття дає можливість порушувати вимоги до алгоритмів AE, тобто у цьому випадку автентичність зберігається, а конфіденційність мінімально зменшується шляхом знання відомих параметрів алгоритму автентифікації. Алгоритми MRAE дозволяє противнику повторно використати Nonce для шифрування повідомлення тим самим ключем. Наприклад, противник може повторювати *nonce* та має доступ до оракула шифрування та розшифрування. Шифруючий оракул повертає будь-які біти, окрім *nonce*, AD, BT, тоді противник знає про виданий результат. Оракул розшифрування може видати ?, коли *nonce*, AD, BT вже відомі для правильного розшифрування. Також MRAE дозволяє випускати неперевірені повідомлення, тобто ці повідомлення будуть випущені, якщо тег не пройшов перевірку, навіть не дивлячись на помилку.

Автентифіковане шифрування з асоційованими даними (AEAD)

Поштовхом до розробки AEAD стало те, що мережеві протоколи,

крім основного тексту, містять поля, наприклад адреси, порти, номера протоколу. Вони мають бути автентифіковані, але не зашифровані для коректного функціонування мережі або системи. В наслідок цього виникла потреба у дешевому та ефективному способі, який забезпечить правильне функціонування мережевих пакетів. Саме автентифіковане шифрування з асоційованими даними забезпечує автентичність цих заголовків. У роботі [2] був введений термін (AEAD).

Означення 1.8. Асоційовані дані — це дані (заголовки), які є автентифікованими, але не конфіденційними. Цілісність цих даних необхідно перевіряти разом з конфіденційними.

Означення 1.9. Автентифікований шифр з асоційованими даними — це сімка виду:

$\Xi = < \mathcal{M}, \mathcal{K}, \mathcal{C}, \mathcal{A}, \tau, \mathcal{NE}, \mathcal{D} >$, де

$\mathcal{M}$ — множина повідомлень;

$\mathcal{K}$ — множина ключів;

$\mathcal{C}$ — множина шифротекстів;

$\mathcal{N} - nonce$

$\mathcal{A}$ — множина не конфіденційних асоційованих даних;

τ - тег;

$\mathcal{E} : \mathcal{M} \times \mathcal{K} \times \mathcal{A} \times \mathcal{N} \to \mathcal{C} \times \tau \times \mathcal{A}$ — шифруюче перетворення;

$\mathcal{D} : \mathcal{C} \times \tau \times \mathcal{K} \times \mathcal{N} \times \mathcal{A} \to \mathcal{M} \times \tau \times \mathcal{A}$ — розшифровуюче перетворення.

Існують два способи перетворення схеми AE в AEAD, які не підтримують асоційовані дані: 'крадіжка' nonce та 'переміщення' шифротексту. Крадіжка 'nonce' полягає у тому, що певна n-бітна частина може бути використана для асоційованих даних. Наприклад, схема AE використовує n-біт nonce, а певний додаток використовує $n^{'}$-біт, при цьому $n^{'} < n$, тоді кількість біт на асоційовані дані дорівнює різниці n та $n^{'}$ біт. При цьому перетворенні є обмеження на розмір асоційованих даних, але адреса деяких Інтернет — протоколів використовують малу кількість біт для заголовків. Щодо 'переміщення' шифротексту, то

спочатку шифрується повідомлення за алгоритмом AE та застосовується певна ключова функція ($F : K \times \{0,1\}^* \to \{0,1\}^\tau$) до рядка будь-якої довжини. Для отримання шифрованого тексту в AEAD використовується сума (XOR) τ лівих бітів τ ключової функції, яка застосовується до бітового рідка будь-якого розміру та правих бітів τ, отриманих в результаті шифрування повідомлення в схемі AE.

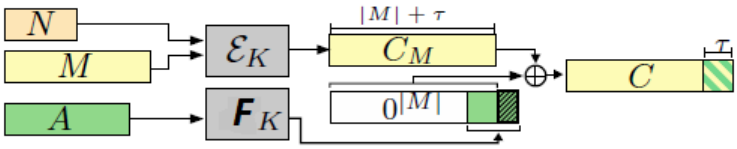

Рисунок 1.5 – Схема 'переміщення' шифротексту

Схеми побудови AEAD на основі блокового шифру та MAC. Реалізувати AEAD можна двома способами: $AD \subseteq \{0,1\}^*, N = \{0,1\}^n, P \subseteq \{0,1\}^*, F : K \times \{0,1\}^* \to \{0,1\}^\tau$:

1) Шифрування — потім — MAC: За допомогою nonce відбувається шифрування $C = \mathcal{E}_K(P)$ потім $T = F_{K'}(AD, C, N)$;

2) MAC — потім — шифрування: Генерується тег за допомогою P, AD, N, потім шифрування P з використанням тегу та N. $C = \mathcal{E}_K(P, T)$, де $T = F_{K'}(AD, P, N)$.

1.2 Загальна характеристика конкурсу CAESAR

27 січня 2014 року було оголошено заклик до подання заявок на конкурс CAESAR [3]. Вже у березні цього ж року на конкурс було подано 57 схем аутентифікованого шифрування. Були висунуті вимоги прийняття участі до учасників даного конкурсу, а саме до будови, безпеки та наявності певних виконання властивостей схем автентифікованого шифрування. Учасників конкурсу можна також класифікувати за їх архітектурою [4]: побудовані на основі блокових шифрів; на основі

потокових шифрів; на основі без ключових перестановок; на основі геш-функцій; на основі інших примітивів. Дане змагання відбулося в три раунди: перший тривав з березня 2014 року по липень 2015 року, другий — з липня 2015 року по серпень 2016 року, третій — з серпня 2016 року по березень 2018 року, фіналістів визначали протягом року — з березня 2018 року по березень 2019 року. Саме 20 березня 2019 року було оголошено фіналістів всесвітнього конкурсу CAESAR. Ними стали алгоритми автентифікованого шифрування ASCON, ACORN, AEGIS-128, OSB, Deoxis, COLM. Переможці були класифіковані на основі їх будови: додатки з обмеженими ресурсами (ASCON, ACORN), додатки з високою продуктивністю (AEGIS-128, OSB), підвищеної стійкості(Deoxis, COLM). Особливості будови схем фіналістів зазначені у таблиці 1.2.

Таблиця 1.1 – Учасники конкурсу CAESAR

++AE	CBA	JAMBU	SCREAM
ACORN	CBEAM	Joltik	SHELL
AEGIS	CLOC	Julius	SILC
AES-CMCC	Deoxys	Ketje	Silver
AES-COBRA	ELMD	Keyak	STRIBOB
AES-COPA	Enchilada	KIASU	Tiaoxin
AES-CPFB	FASER	LAC	TriviA-ck
AEZ	HKS	Marble	Wheesht
Ascon	HS1-SIV	McMambo	YAES
AVALANCHE	ICEPOLE	Minalpher	
Calico	iFeed	MORUS	
NORX	OCB	OMD	
PAES	PANDA	π-Cipher	
POET	POLAWIS	Prost	
PRIMATEs	Raviyoyla	Sablier	

Таблиця 1.2 – Фіналісти конкурсу CAESAR

Учасники	Конструкція та особливості
ASCON	На основі спонжу. Онлайн та оберненість.
ACORN	На основі потокового шифру. Використовує ЛРЗЗЗ,онлайн, можливість розпаралелювання, оберненість
AEGIS-128	Визначений примітив. Можливість розпаралелювання шифрування, онлайн, онлайн.
OSB	На основі блок-шифру. Можливість розпаралелювання, онлайн, покроковість асоційованих даних.
Deoxis-||	На основі блок-шифру. Можливість розпаралелювання, онлайн.
COLM	Результат об'єднання двох учасників конкурсу AES-COPA та ELmD. На основі блок-шифру. Можливість розпаралелювання, онлайн.

1.2.1 Функціональні вимоги до учасників

Схема аутентифікації повинна мати п'ять входів та два виходи. Алгоритм шифрування повинен приймати на вхід:

1) відкритий текст змінної довжини(більше ніж 2^{16});

2) асоційовані данні змінної довжини(більше ніж 2^{16});

3) секретного повідомлення фіксованої довжини;

4) публічне повідомлення фіксованої довжини;

5) ключ фіксованої довжини.

Результатом алгоритму шифрування має бути шифротекст та тег підтвердження. Повинна бути представлена таблиця довжин характеристики, що мають фіксовану довжину. Це необхідно для того, щоб будь-хто міг реалізувати шифр-текст не маючи більше інформації, ніж зазначеної у таблиці.

1.2.2 Вимоги безпеки

За вимогами конкурсу учасникам потрібно надати таблицю, у якій зазначено кількість біт для конфіденційності відкритого тексту та цілісності асоційованих даних.

1.2.3 Властивості схем автентифікованого шифрування

Можливість розпаралелювання. Операція шифрування є паралельною, якщо обробка j-того вхідного блоку не залежить від обробки j-того блоку. При паралельному шифруванні або дешифрування одночасно обчислюється кілька блоків.

Одно прохід.Необхідно один раз зашифрувати відкритий текст для забезпечення аутентифікованого шифрування.

Онлайн. Часто використовуються криптографічні пристрої з обмеженою пам'яттю, тому у повному обсязі шифротекст не може зберігатися.Існують онлайн — шифри, які шифрують довільну кількість блоків відкритих текстів і виводять блоки шифротексту, які залежать тільки від попередніх блоків відкритого тексту. Аутентифіковане шифрування може відбуватися без знання довжини відкритого тексту та асоційованих даних.

Оборотність.Схема аутентифікованого шифрування називається зворотною, якщо вона не вимагає ані прямої, ані зворотної операції її основного примітиву, наприклад, як це вимагає функція дешифрування блокового шифру.

Покроковість аутентифікованого шифрування. Користувач може змінити блок відкритого і відповідний блок шифрованого тексту без необхідності повторного шифрування всього повідомлення.

Покроковість асоційованих даних. Схема потребує покрокових аутентифікованих асоційованих даних, якщо зміна блоку та фінального

кроку потребує оновлення.

Проміжні теги.Включення проміжних тегів аутентифікації через однакові інтервали в шифротексті дозволяє завчасно відкрити способи підробки повідомлення.

1.2.4 Класифікація учасників конкурсу на основі архітектури

Учасників даного конкурсу можна поділити на 7 класів на основі використаного ними криптографічного примітиву:

На основі блок-шифру. 24 учасники використовують блок-шифр як внутрішній примітив, наприклад Deoxys, AEZ, Cloc, Kiasu, OCB, POET, SCREAM, SHELL і т.д. Деякі учасники використовують певні режими блок-шифру, наприклад CTR (3 учасників), ECB (2 учасників), CBC (1 учасник).

На основі потокового шифру. 7 учасників використовують потоковий шифр, який вже існу, наприклад ACORN та MORUS.

На основі перестановки. Наприклад, без ключову перестановку або перестановку на основі конструкцій, що існують.

На основі губки. Дев'ять учасників використовують без ключову перестановку у губковому режимі роботи, наприклад Ascon, Keyak та NORX.

На основі функції стиснення. Один кандидат (OMD) використовує функцію стиснення з геш-функцій $SHA256$ та $SHA512$.

На основі інших примітивів. Троє учасників використовують примітив, який не був згаданий вище, наприклад AEGIS.

Не використовується примітив. Тільки один учасник (POLAWIS) не використовує типовий симетричний примітив.

1.2.5 Класифікація на основі режимів роботи

Алгоритми, які використовують базовий криптографічний примітив для забезпечення безпеки та автентичності називають режимом роботи. Учасники на основі блокового шифру використовують такі режими роботи. Наступні учасники CAESAR використовують 'налаштований' блоковий шифр, як внутрішній примітив: Silver, SCREAM, KIASU≠, Joltik≠, iSCREAM, Deoxys≠, OCB.

Висновки до розділу 1

У даному розділі проаналізовані літературні джерела за даною тематикою дослідження, наведені основні теоретичні означення, що стосуються автентифікованих шифрів, вимоги до учасників конкурсу CAESAR, наведена класифікація учасників на основі їх архітектури. Також зазначені учасники, які використовують налаштований блоковий шифр, як внутрішній примітив.

2 НАЛАШТОВАНІ БЛОКОВІ ШИФРИ, ПРЕДСТАВЛЕНІ У КОНКУРСІ CAESAR ТА ЇХ КРИПТОАНАЛІЗ

У розділі розглянуто загальну схему побудови налаштованих блокових шифрів, а також конкретні конструкції, які були застосовані у деяких схемах учасників конкурсу CAESAR. Детально розглянуто схеми учасників конкурсу, а саме аутсайдерів Silver, Kiasu та переможця OCB та застосування інтегрального криптоаналізу до схем аутсайдерів.

2.1 Налаштовані блокові шифри

Блокові шифри є найпопулярнішими криптографічними примітивами, які застосовуються в якості базових блоків, що забезпечують безпеку та автентичність. У автентифікованому шифруванні базовий блоковий шифр використовується і для шифрування, і для створення MAC за допомогою різних режимів роботи. Якщо при шифруванні і (або) створенні тегу використовується один і той самий ключ до всіх блоків, то це надає зловмиснику можливість маніпулювання блоками (наприклад, їх переставляти) та застосування атак зі зв'язаними ключами. Крім того, ключовий розклад є зазвичай складною процедурою, то генерування кожного разу нового ключа критично знижує ефективність шифрування. Тому з'явилися алгоритми, у яких на вхід блокового шифру крім ключа та відкритого тексту подається додатковий параметр - твік (tweak), який «налаштовує» процес шифрування так, що блоки шифруються різними шифрами з одного сімейства. Термін «налаштований (tweakable) блоковий шифр» був запропонований у 2002 році Лісковим, Ривестом та Вагнером у роботі [5], де була також показана можливість застосування таких шифрів у схемах автентифікованого шифрування. Однією з цілей застосування твіку є

створення безпечних режимів роботи на основі блокового шифру. Прикладом таких режимів роботи є TAE (Tweakable authenticated encryption), XE (XOR-Encrypt) та XEX (XOR-Encrypt-XOR).

Означення 2.1. 'Налаштований' блоковий шифр - це відображення виду: $E : K \times T \times P \to C$, де

$K \in \mathcal{K} \subseteq \{0,1\}^k$ - множина ключів;

$P \in \mathcal{P} \subseteq \{0,1\}^n$ - множина ключів;

$C \in \mathcal{C} \subseteq \{0,1\}^n$ - множина шифротекстів;

Лісков, Рівест та Вагнер зазначили, що зміна твіку має бути менш затратною, ніж зміна ключа, зміна твіку має бути послідовною. Зміна значення твіку приводе до повторного виклику блокового шифру, тому що ключ залежить від твіку. У роботі [6] зазначено, що дане твердження нелогічне, тому що супротивник контролює твік, а доступ до ключа у нього обмежений. Тому твік та ключ повинні розглядатися однаково. Для ефективної реалізації налаштованого блокового шифру зміна твіку повинна відбуватись без повторного виклику процедури шифрування. Тому J. Jean із співавторами розробили структуру Tweakey [6].

Структура Tweakey

Конструкція *Tweakey* застосовується для розробки налаштованих блокових шифрів, у яких ключ та твік в основному трактуються як один об'єкт під назвою *tweakey*. Структура дозволяє додавати твік (майже) будь-якої довжини до блокового шифру з послідовним введенням ключів та розширити ключовий простір до (майже) будь-якого розміру. Кожні *subtweakey* створені шляхом застосування однієї і тієї перестановки. У цій конструкції використовується *tweakey* розклад замість ключового розкладу у блокових шифрах. Структура складається з subtweakey g, перестановки внутрішнього стану f та функції оновлення tweakey стану h. Шифротекст обчислюється з відкритого тексту: на кожній ітерації застосовується перестановка внутрішнього стану f та subtweakey ксориться з внутрішнім станом кожного раунду. Клас налаштованих блокових шифрів позначається як $TK - p$ та вводиться, коли розмір

шифрування $(p \times n)$ біт. Клас налаштованих блокових шифрів $TK-2$, коли n-біт ключ та n-біт твік. Також у роботі [6] зазначено підклас $Tweakey$ - це конструкція STK. У STK внутрішні стани та стани $tweakey$ розділені на $\frac{n}{c}$ та $\frac{pn}{c}$, де c-біт показник $GF(2^c)$. Функція h поділяється на дві функції h^* та α_j, де h^* функція перестановки для c та α_j нульовий коефіцієнт, які множиться на кожний c-біт показник над полем $GF(2^c)$. Функція g являє собою XOR pn біт станів. Структура Tweakey застосовується у схемах автентифікованого шифрування конкурсу CEASAR: Silver та Kiasu, Joltic.

Застосування конструкцій XE та XEX

Одним із важливих методів побудови налаштованих блокових шифрів є побудова XE та XEX конструкцій. Ці конструкції застосовується у більшості учасників конкурсу CAESAR, наприклад AES-COPA, ELmD, OCB. XE та XEX конструкції перетворюють блоковий шифр у налаштований блоковий шифр, де множина твіків τ - це простір вигляду: $\tau = \{0,1\}^n \times \mathcal{I} \times \mathcal{J}$, де $\mathcal{I}$ - набір кортежей великих чисел, $\mathcal{J}$ - набір кортежей малих чисел. Метод ефективний, коли твіки виникають послідовно. Нехай, $N \in \{0,1\}^*$, $i \in \mathcal{I}$, $j \in \mathcal{J}$. Тоді більшість твіків $(N, i_1, \ldots, i_k)$ являються приростом іншого твіку, якщо один із i збільшився, а інші залишаються незмінними .

XE конструкція: $\widetilde{E}_K^{N,i} = E_k(M + \Delta)$

XEX конструкція: $\widetilde{E}_K^{N,i} = E_k(M + \Delta) + \Delta$, де $\Delta = f(\mathcal{N})$. Метою Δ є 'маскування' блоку відкритого тексту. Існують різні способи обчислення Δ. Одним із таких способів є обчислення $\Delta = x^i \cdot L$, де $L = \mathcal{E}_k(\mathcal{N})$, $\cdot$-операція множення у полі F_{2^n} та x-генератор поля F_{2^n}. Інший спосіб - це маски у вигляді кодів Грея $\varphi_i \cdot L$, де $\varphi_i = i \oplus (i >> 1)$.

Безпека налаштованих блокових шифрів $\mathcal{D}$ (distinguisher) - алгоритм, у якого є можливість надати доступ до запиту одного з оракулів або $\mathcal{O}$, або $\widetilde{\mathcal{O}}$ та на виході мати один біт. Ця характерна перевага для $\mathcal{O}$ та $\widetilde{\mathcal{O}}$ визначається як: $Adv(\mathcal{D}) = |P(\mathcal{D}^{\mathcal{O}} \Rightarrow 1) - P(\mathcal{D}^{\widetilde{\mathcal{O}}} \Rightarrow 1)|$

Нехай, $\widetilde{E}$ - налаштований блоковий шифр. Для будь-яких $K \in \mathcal{K}$,

$t \in \mathcal{T}$ та $\widetilde{E}_K^T(\cdot)$ - перестановка на $\{0,1\}^n$. Маємо $P(n)$ - множина всіх n-бітних перестановок, $P(\mathcal{T}, n)$ множина всіх відображень із $\mathcal{T}$ в n - бітну перестановку. Нехай π - будь-яка перестановка з множини $P(\mathcal{T}, n)$, позначимо $\pi(\mathcal{T}, \cdot)$

Означення 2.2. Нехай $\mathcal{A}$ - противник з доступом до шифруючого оракула. Перевага противника $\mathcal{A}$ над $\widetilde{E}$:

$$Adv_E(\mathcal{A}) = P(\mathcal{A}^{\mathcal{E}_k(\cdot,\cdot),\mathcal{E}_k^{-1}(\cdot,\cdot)} \Rightarrow 1) - P(\mathcal{A}^{\pi(\cdot,\cdot),\pi^{-1}(\cdot,\cdot)} \Rightarrow 1)$$

2.2 Основи інтегрального криптоаналізу

У 1997 році знаменитими криптографами Кнудсеном, Рюменом та Даменом був розроблений блоковий шифр Square [7], який є попередником шифру RIJNDAEL і, відповідно, стандарту AES. Після дослідження будови шифру була описана атака на шифр Square. Пізніше Кнудсен та Вагнер узагальнили та покращили атаку, яка отримала назву інтегральний криптоаналіз [8]. Інтегральний криптоаналіз є одним із найпотужніших методів криптоаналізу, який застосовується на блокові шифри. Опишемо більш детально дане дослідження.

Нехай (G, +) - скінченна абелева група порядку k. Розглянемо декартів добуток n таких груп: $G^n = \prod_{i=1}^{n} G$, де кожен елемент представлений як вектор з n компонент: $v_i \in G : \mathbf{v} = (v_1, ..., v_n)$. Операція додавання векторів визначена покомпонентно: $\forall \mathbf{u}, i, \mathbf{v}, \mathbf{w} \in G^n : u_i + v_i = w_i$.

Нехай $\Lambda \subseteq G^n$ - мультипідмножина.

Означення 2.3. Інтеграл по мультимножині векторів Λ - це сума по всіх векторах даної множини:

$$\int \Lambda = \sum_{\mathbf{v} \in \Lambda} \mathbf{v},$$

де операція додавання векторів відбувається як визначено вище.

Зазвичай в інтегральному криптоаналізі G — адитивна група $GF(2^r)$, а множина Λ є деякою множиною блоків шифрованого тексту.

Атака полягає в тому, що криптоаналітик намагається передбачити значення інтегралу після певного раунду шифрування. Ми будемо розглядати три можливі випадки, які є важливими для нас. Розглянемо деякий фіксований індекс $0 \leq i \leq n$:

1) i-ті компоненти однакові: $v_i = c$ для всіх векторів $\mathbf{v} \in \Lambda$;

2) Всі компоненти різні: $\{v_i : \mathbf{v} \in \Lambda\} = G$;

3) Сума компонент: $\sum\limits_{\mathbf{v}\in\Lambda} v_i = c^{'}$,

де $c^{'}$, $c \in G$ - деякі фіксовані значення.

Розглянемо випадок, коли кількість векторів у вибраній мультипідмножині Λ збігається з кількістю елементів у групі G. Тоді якщо всі i-ті шифротексти рівні, то очевидно, що їх сума дорівнює нейтральному елементу групи G. У другому випадку при $G = GF(2^r)$ і $|\Lambda| = 2^r$ сума i-тих компонент у множині Λ також дорівнює 0. Тобто $\sum\limits_{\mathbf{v}\in\Lambda} \mathbf{v} = \sum\limits_{g\in G} g = 0$(добре відомий факт з теорії скінчених полів).

Таким чином, у кожному з трьох випадків можна передбачити значення інтегралу по множині Λ.

Нехай, у шифрі обчислюється $u_i + v_i = w_i$, де u_i, v_i, w_i — проміжні значення, і інтеграл передбачає, що u_i, v_i мають властивості 1), 2) або 3). Тоді $\sum\limits_{\Lambda} w_i = \sum\limits_{\Lambda} u_i + \sum\limits_{\Lambda} v_i$ також відома. Зокрема всі u_i однакові, а v_i — різні, то w_i — також різні і навпаки. Тепер розглянемо нелінійне перетворення $f : v_i = f(u_i)$. Зрозуміло, якщо u_i однакові, то v_i — також будуть однаковими. Якщо ж f бієкція, то у випадку різних u_i різними будуть і v_i.

2.2.1 Застосування інтегрального криптоаналізу до AES — подібних шифрів

Розглянемо як інтегральний криптоаналіз застосовується до AES з блоком 128 біт, де кожен блок представлений у вигляді матриці з байтів розміру 4×4. У термінах пункту 2.2 G — адитивна група $GF(2^8)$ (тобто $\mathbf{u}_i$ — це байти), $n = 16$ (тобто вектори $\mathbf{v}$ — це блоки).

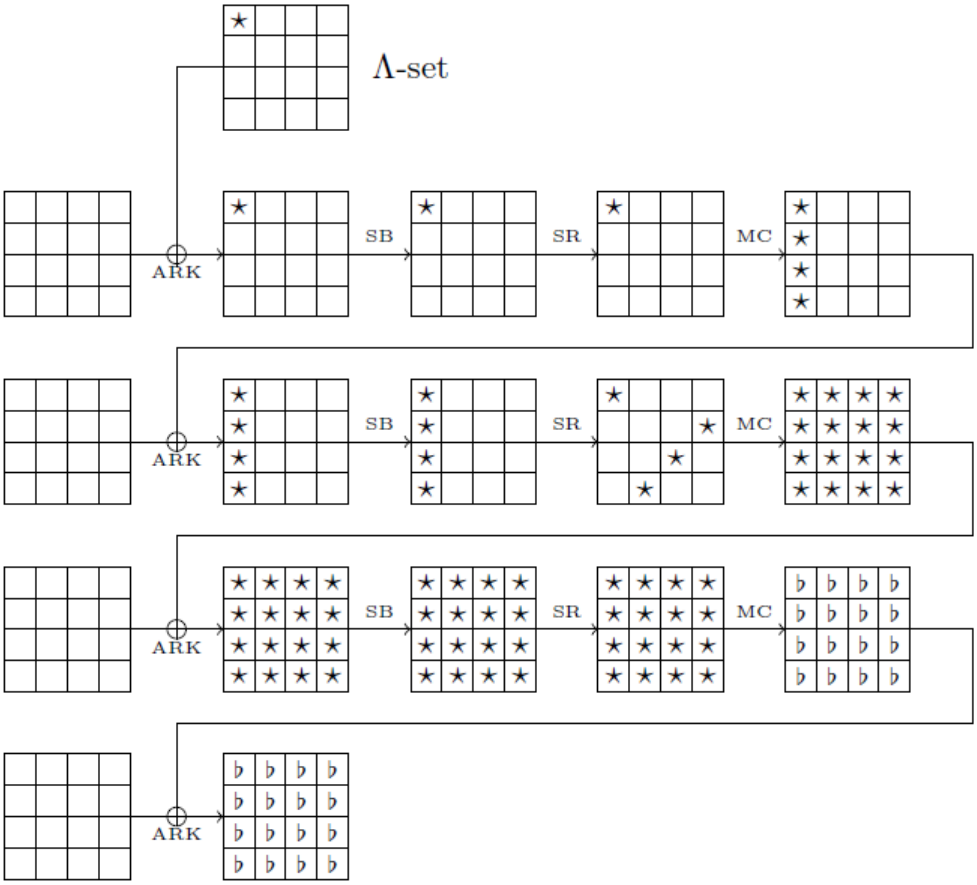

Рисунок 2.1 – Схема чотири раундової інтегральної атаки

Нехай, Λ — множина, яка складається з 256 блоків: $\Lambda = \{ v_j, 0 \leq \mathrm{j} \leq 256 \}$. Кожен блок v_j складається з 16 байтів, де байт $x_i(j)$, $1 \leq i \leq n$ стоїть на i-тій позиції в j-тому блоці v_j. Якщо $x_i(j)$ при $0 \leq j \leq 256$ приймає всі можливі значення, то позицію i в блоці будемо називати

активною та позначимо як $*$. Якщо ж усі $x_i(j) = c$ при $0 \leq j \leq 256$, то позицію i будемо називати сталою і позначимо як C (constant). Якщо ми можемо передбачити суму бітів на i-тій позиції по всіх блоках з Λ, то таку позицію будемо позначати як b. Тепер розглянемо як змінюється позиція байта при перетвореннях, що відбуваються на одному циклі AES. Оскільки операції SubBytes(SB) та AddRoundKey(ARK) діють на кожен байт окремо і незалежно від інших байтів, тобто раундовий ключ однаковий для кожного блоку з Λ, і перетворення байтів при цьому є бієкцією, то позиції 'C' і позиції '*' такими й залишаються. Операція ShiftRow(SR) просто зміщує байти, тому й активні та сталі байти при цьому змінюються відповідно. А от MixColumn(MC) перемішує стовпці, при цьому байт на активній позиції впливає на всі інші байти та через те, що перетворення кожного байта при цьому є оборотним, а отже бієктивним, то всі позиції в стовпчику стануть активними. Застосування MC до стовпчика, у якому всі байти активні, не обов'язково дає стовпчик з усіма активними байтами, проте суму байтів по множині Λ можна передбачити: вона дорівнює нулю. Перетворення множини Λ протягом трьох циклів шифрування можна побачити на рис. 2.1. Відмітимо, що зазначених інтегралів існує 16 в залежності від позиції активного байта на початку.

Описану властивість шифру AES називають інтегральною властивістю. Її можна застосовувати до знаходження ключа у 4-раундовому AES. Так як останній раунд (у даному випадку — четвертий) не містить операції MixColumn, то перебираючи один байт раундового ключа k_4, відповідний байт шифрованого тексту можна частково розшифрувати, отримавши його значення в кінці третього раунду. Якщо сума байтів на відповідному місці дорівнює нулю, то байт ключа був вірним.

Атаку можна розповсюдити на 5 раундів, використовуючи той самий інтеграл. Для цього потрібно перебрати один байт ключа на 5-му раунді та чотири байти на четвертому — всього 5 байтів. Існує також

спосіб розширити атаку і на 6 раундів використовуючи 2^{32} блоків BT на першому раунді.

2.3 Опис алгоритму автентифікованого шифрування Silver

Silver — один з низки учасників конкурсу CEASAR, побудованих на основі налаштованого AES. Автори цієї схеми — Пенацці та Монтес [9] — розробили особливу, досить складну систему введення твіку, яка відрізняється від способів налаштування в інших схемах автентифікованого шифрування на базі налаштованих блокових шифрів. Проте цей алгоритм виявився недостатньо стійким і вибув з конкурсу після першого етапу. До нього було застосовано низку криптоатак, які базуються на інтегральній властивості AES. Властивості цих атак та їх складність і трудомісткість наведені у таблиці 2.1 [10].

Таблиця 2.1 – Складності атак на Silver

Модель	Кількість раундів	Тип атаки	Складність
Nonce respecting	4	Підробка AD	2^{79}
	8	Підробка AD	2^{111}
Nonce repeating	Всі (11)	Підробка BT	$2^{49.46}$
	Всі (11)	Відновлення BT	1
	8	Відновлення ключа	2^{111}

Під складністю тут розуміється кількість звернень до оракула. Зазначимо, що атака відновлення відкритого тексту при одному зверненні до оракула (зашифрування) можливе лише для BT, що складається з одного неповного блоку та при відсутності асоційованих даних. Ця атака розглянута у пункті 2.3.2. Також детально розглянута криптоатака підробки AD при чотирьохраундовому AES (п. 2.3.1).

Silver – це схема AE, яка має в якості внутрішнього криптографічного примітиву налаштований AES - 128. Silver має 4 входи:

128-бітний несекретний номер повідомлення $\mathcal{N}$ (nonce), 128-бітний секретний ключ $\mathcal{K}$, несекретні асоційовані дані $\mathcal{A}$ (можливо відсутні) та відкритий текст $\mathcal{P}$. На виході отримується ШТ $\mathcal{C}$ та 128-бітний тег $\mathcal{T}$. Алгоритм розшифрування з четвірки $(\mathcal{N}, \mathcal{A}, \mathcal{C}, \mathcal{T})$ отримує ВТ $\mathcal{P}$, якщо тег $\mathcal{T}$ пройшов перевірку і $\perp$ у протилежному випадку. Уведемо деякі позначення: b_P та b_A - довжини у байтах ВТ $\mathcal{P}$ і АД $\mathcal{A}$, $g = b_A \parallel b_P$ - перетворює ці два числа у 128-бітний вектор. $\oplus$ - означає XOR, $+$ - додавання за модулем 2^{64}, яке виконується для двох половинок 128-бітних величин незалежно. Ліву половину (старші 64 біти) 128-бітної величини x будемо позначати x^L, а 64 молодших біти - x^R.Таким чином, $x = x^L \parallel x^R$, де $\parallel$ - конкатенація. $\overline{x}$ - величина x, у якій молодший біт примусово покладається 1, тобто $\overline{x} = x \vee 1$.

Алгоритм шифрування блоку тексту E_K складається з 11 циклів, 128-бітового AES, налаштування якого полягає у специфічному способі генерування раундових ключів. А саме, раундові ключі для шифрування блоку ВТ генеруються на основі ключа $\mathcal{K}$, nonce $\mathcal{N}$ та твіку, який залежить від позиції блоку в тексті. Спочатку шифрується nonce за допомогою спеціального алгоритму AES - 128 з ключем $\mathcal{K}$: $\alpha = AES_K(N)$. Потім за допомогою ключового розкладу AES - 128, який ми позначимо KS (key schadule) формуються дві послідовності: $(k_0, \ldots, k_{10}) = KS(K)$ і $(\alpha_0, \ldots, \alpha_{10}) = KS(\alpha)$. Циклові ключі для 11 раундів шифрування мають вид:

$K_j = k_j \oplus \alpha_j$ для $j = 0, 2, 3, 4, 6, 7, 8, 10$

$K_j = k_j + \alpha_j \oplus (\alpha_j + i\gamma)$ для $j = 1, 5, 9$

Тут i — номер блоку ВТ або AD, який шифрується. Величина γ відрізняється при шифруванні ВТ та AD. Для ВТ - $\gamma = \gamma_P = \overline{\alpha_9^L} \parallel \alpha_9^R$, а при обробці АД - $\gamma = \gamma_A = \overline{\alpha_9^L} \parallel 0^{64}$.

Отже, $i\gamma$ є твіком, який залежить від $\mathcal{N}$ та номеру блоку, що шифрується.

Створення MAC для AD. Асоційовані дані $\mathcal{A}$ поділяються на 128-бітові блоки $A_1, \ldots, A_t$ у випадку неповного останнього блоку він

доповнюється бітами 10...0 з необхідною кількістю нулів (будемо називати такий спосіб доповнення блоку 10^* - доповненням). Далі AD шифруються налаштованим AES з твіком $i\gamma_A$. Шифровані блоки утворюють $\sum_{i=1}^{t} A_i = \sum_A$, де під $\sum$ розуміється XOR. Якщо останній блок неповний, то після доповнення він шифрується з нульовим твіком ($\gamma_A = 0$). Алгоритм обчислення контрольної суми $\sum_A$ зображений на рисунку 2.2.

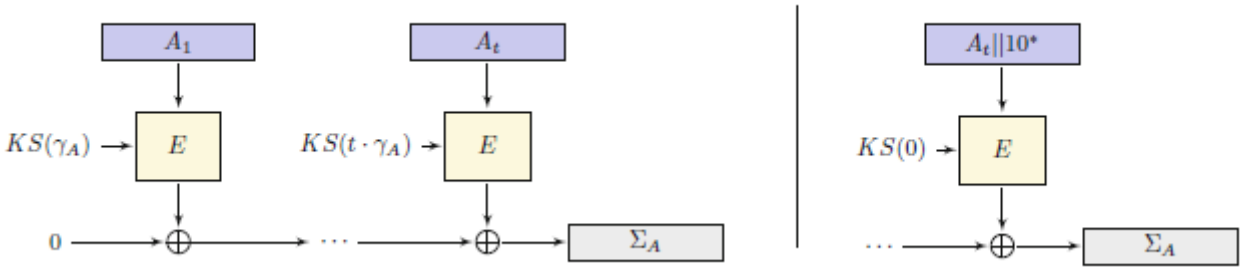

Рисунок 2.2 – Обчислення контрольної суми $\sum_A$

Шифрування. Позначимо через s кількість 128-бітових блоків відкритого тексту P, останній, можливо, неповний. Блок P_i шифрується налаштованим алгоритмом AES з твіком $i\gamma$ так, як описано раніше, C_i - відповідний блок ШТ. Контрольна сума $\sum_P$ - це XOR всіх блоків ВТ, якщо останній блок P_s повний. Якщо ж P_s - неповний, то він шифрується особливим чином (дивись праву частину рисунка 2.3). Нехай P_s має $0 < l < 16$ байтів. Тоді він ксориться з 'маскою' μ, що є результатом шифрування $b^P \parallel b^P$ з твіком $s\gamma$ (при цьому P_s доповнюється нулями). Останній байт утвореної суми замінюється на двійковий запис числа l. Отриманий 128-бітовий вектор шифрується з твіком $(s+1)\gamma$. Результат шифрування позначимо $\sum_P^{'}$. У результаті шифрування обчислюється також контрольна сума $\sum_C = \bigoplus_{i=1}^{s}(c_i + \alpha + i\gamma)$.

Генерування тегу. Підраховується XOR чотирьох контрольних сум: $\sum_A \oplus \sum_P \oplus \sum_C \oplus \sum_p^{'} = \sum$. Тег T є результатом шифрування $\sum$ з твіком $g = b^A \parallel b^P$ і раундовими ключами, що переставлені згідно з перестановкою

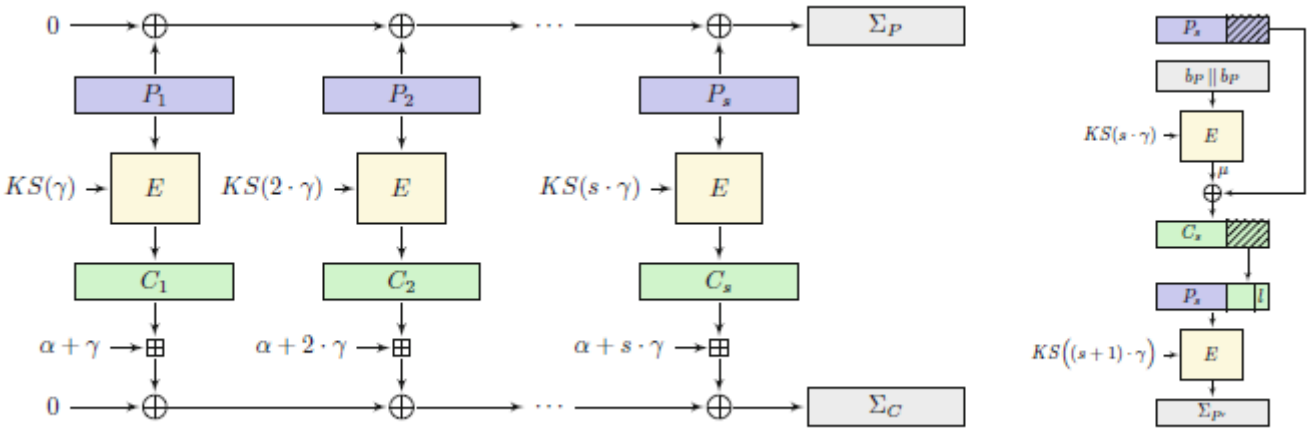

Рисунок 2.3 – Шифрування

$\pi = (5, 4, 6, 10, 55, 69, 578)$.

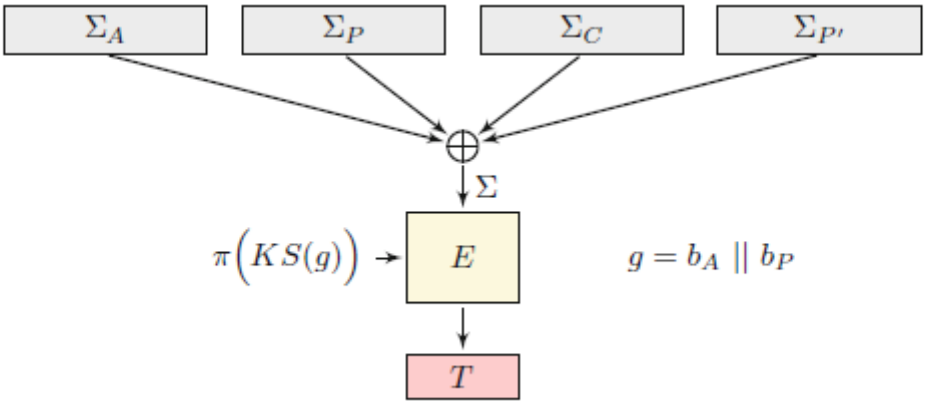

Рисунок 2.4 – Створення тегу

2.3.1 Атака підробки асоційованих даних

Опишемо атаку на алгоритм Silver з урізаною кількістю циклів, якщо зловмисник не може повторювати nonce [10]. В атаці використовується інтегральна властивість AES, описана у розділі 2.2. Метою атаки є знаходження 256 блоків AD $A = (A_1, \ldots, A_{255}, A_0)$ таких, що $\sum_A^{'} = 0$. Це дозволяє підмінити A іншими AD $A^{'}$ так що $\sum_A^{'} = 0$

тобто підробити AD. Спочатку розглянемо атаку на 4-раундовий Silver.

У пункті 2.3 була введена величина $\gamma_A = \overline{\alpha_9^L} \parallel 0^{64}$. У $\overline{\alpha_9^L}$ останній біт покладається рівним 1. Отже, з імовірністю $2^{-63}\gamma_A = 0^{63}1 \parallel 0^{64}$. У такому разі послідовність твіків при неповному останньому блоці AD з урахуванням того, що неповний блок шифрується з нульовим твіком (див. рис 3.2) має вид:

$\gamma_A = 0^{56}00000001 \parallel 0^{64}$,

$2 \cdot \gamma_A = 0^{56}00000010 \parallel 0^{64}$,

$3 \cdot \gamma_A = 0^{56}00000011 \parallel 0^{64}$,

$\dots$

$256 \cdot \gamma_A = 0^{56}11111111 \parallel 0^{64}$,

$0 = 0^{56}00000000 \parallel 0^{64}$

Додатково вимагається, щоб вісім останніх бітів a^L дорівнювали нулю, тоді при додаванні $\alpha + i\gamma$ не виникає перенесеня. Така подія відбувається з імовірністю 2^{-8}. Отже, з імовірністю 2^{-71} у величинах $\alpha + i\gamma_A$ при $0 \leqslant i \geqslant 255$ біти з 57-го до 63-го (один байт) приймають всі можливі значення.

Виберемо $A = A_1 \parallel \cdots \parallel A_{255} \parallel A_0$ так, що $A_0 = B$, $A_i = B \parallel 10^7$, $1 \leqslant i \geqslant 256$, де B - довільна 120-бітна величина. Це забезпечує однаковість всіх блоків від 1-го до 255-го. Останній, A_0 після 10^* - доповнення буде таким самим. Процес обробки АД $A = A_1 \parallel \cdots \parallel A_{255} \parallel A_0$ зображений на рисунку 2.5.

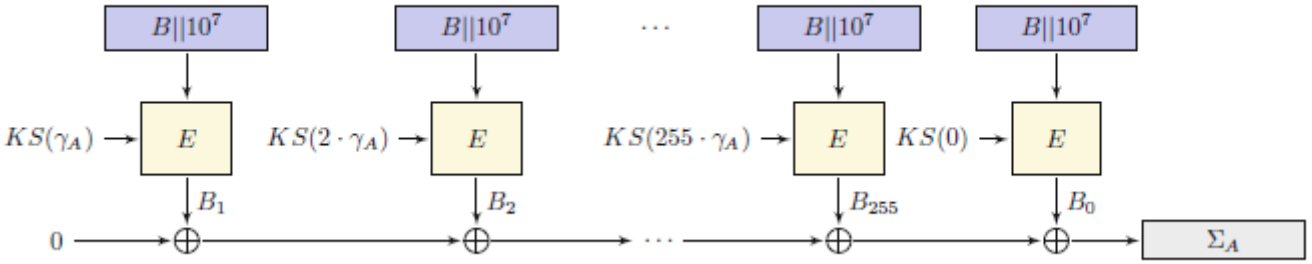

Рисунок 2.5 – Створення тегу

Нагадаємо, що E в такому випадку означає 4-раундовий налаштований AES. Використовуючи інтегральну властивість AES, описану в пункті 2.2, покажемо, що при даному виборі АД $\sum_A = 0$. На рисунку 2.1 зображений процес шифрування одного блоку A_1. Байт позначається як C, якщо він однаковий у цій позиції для одного стану шифру у всіх 256 блоках; позначається як A, якщо він приймає всі можливі значення у 256 блоках і 0, якщо XOR байтів на цій позиції по всіх 256 байтах дорівнює 0.

На вхід подаються однакові блоки $B \parallel 10^7$, про те на початку другого раунду при сумуванні з раундовим ключем $K_1 \oplus (\alpha + i\gamma_A)$ у всіх блоках восьмий байт стане різним. Далі все відбувається згідно з інтегральною властивістю AES, описаною у пункті 2.2, і в результаті отримується $\sum_A = 0$. Отже, можна сподіватись, що з 2^71 звернень з різними nonce до шифруючого оракула, одне дасть в результаті пару $(\mathcal{C}, \mathcal{T})$ із внутрішньою сумою $\sum_A = 0$. Таким чином, зловмисник, взявши AD того ж виду A' з іншою 120-бітною величиною $B' = B$ із тим самим $\mathcal{N}$ отримає ту саму контрольну суму і ту саму пару $(\mathcal{C}, \mathcal{T})$, що є підробною АД.

Атака є nonce-respecting, оскільки не вимагає повторення $\mathcal{N}$, і має складність $2^{71} \cdot 2^8 = 2^{79}$. Цю атаку можна розповсюдити на 8 раундів. Так само, як і в попередньому випадку, ми шукаємо 256 блоків A_i таких, що $\sum_A = 0$. Зазначимо, що тільки в раундах 1 і 5 (з 8) використовується твік. Отже, достатньо знайти 256 блоків A_i таких, щоб на вході 5-го раунду всі байти стану були однаковими у всіх 256 блоках (мали тип C). У [6] показано, що цього можна досягти, вгадавши (перебравши) певні чотири байти першого раундового ключа $k_0 + \alpha_1$. Отже, складність атаки дорівнює $2^{79} \cdot 2^{32} = 2^{111}$. При цьому AD повинні мати такий самий вид, що і в попередній атаці.

Проаналізуємо, які властивості алгоритму Silver дозволили розробити описану криптоатаку. По-перше, твік уводиться безпосередньо у ключ, і після генерування ключа алгоритм працює як звичайний AES.

По-друге, твік використовується лише у трьох раундах з 11, на кожному четвертому раунді. По-третє, останній неповний блок AD шифрується взагалі з нульовим твіком. Ці три особливості дозволяють побудувати атаку підробки AD на 4-цикловий та 8-цикловий Silver з використанням інтегрального аналізу, в саме, інтегральної властивості AES.

2.3.2 Атака відновлення відкритого тексту

Нехай, P — відкритий текст, що складається з одного неповного блоку з 15 байтів. Також вважатимемо, що AD відсутні. Неповний блок шифрується особливим чином (дивись праву частину рисунка 2.3). Спочатку генерується маска μ шляхом шифрування блоку $b_p \parallel b_p$ з секретним ключем і з твіком $s\gamma$, де b_p — довжина ВТ у байтах, s — кількість блоків відкритого тексту. Шифрований текст $C = P \oplus \mu$. Подальші дії щодо генерування тегу не мають значення. Криптоаналітик, отримавши ШТ C з $|C| = 15$, може розшифрувати C, зробивши один запит до оракула щодо зашифрування блоку $b_p \parallel b_p = 15 \parallel 15$ з тим самим $\mathcal{N}$ (Nonce repeating model) із тим самим твіком, бо $s = 1$. Отже, він отримує μ і розшифровує ВТ: $C = P \oplus \mu$.

2.4 Опис алгоритму Kiasu

Одним з учасників конкурсу CAESAR є Kiasu [11]. Kiasu являє собою схему автентифікованого шифрування, де внутрішнім криптографічним примітивом є налаштований блоковий шифр (AES). В основі Kiasu лежить структура Tweakey, яка об'єднує твік та ключ у єдину конструкцію. Tweakey дозволяє генерувати subtweakeys — ключі для кожного раунду, використовуючи ключовий розклад. Опишемо один із варіантів алгоритму Kiasu, а саме Kiasu$\neq$. Kiasu$\neq$ - це режим шифрування, який вимагає забезпечення безпеки, шляхом 'дотримання

nonce' (nonce respecting). Тобто nonce ніколи не може бути використаний для подвійного шифрування з одним і тим самим ключем. Це гарантує, що кожний виклик налаштованого блокового шифру під час шифрування буде мати різні вхідні значення твіку. В основі лежить режим роботи $\Theta CB - 3$ [12]. 64-бітний твік побудований шляхом конкатенації цілого числа від 1 до 5, 32 біт Nonce та цілого значення i, яке показує номер блоку відкритого тексту застосовного при певному шифруванні. Розмір блоку та ключа Kiasu≠ дорівнює 128 біт.

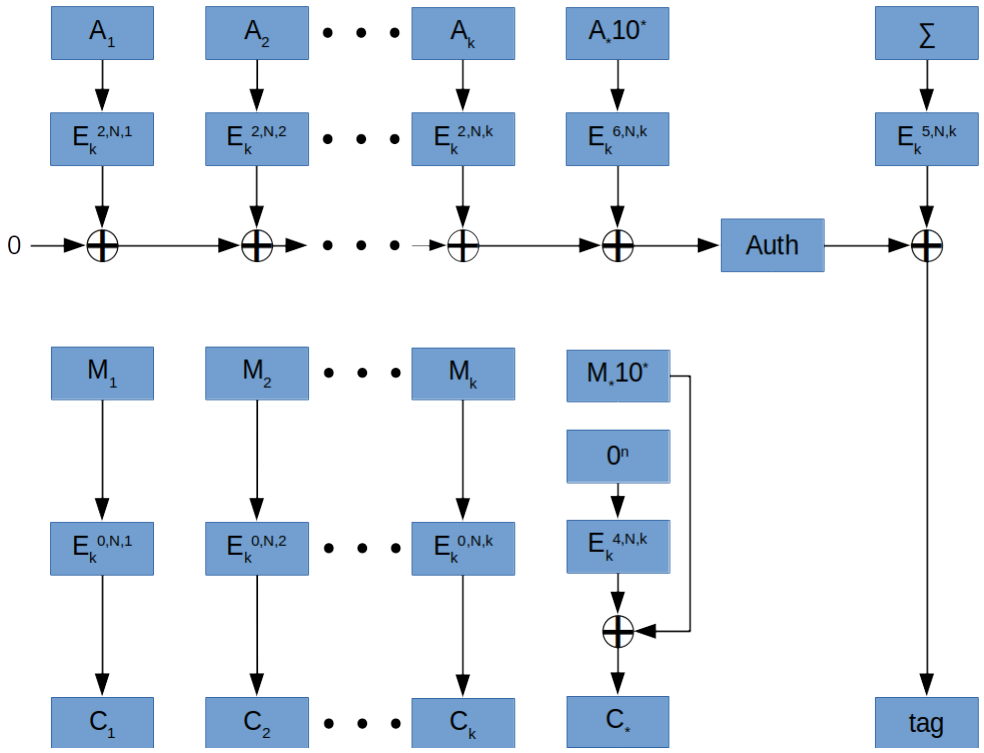

Рисунок 2.6 – Схема автентифікованого шифрування Kiasu≠

Процес шифрування майже такий самий як у 128-бітового AES та складається з 10 циклів. Відмінність полягає в операції AddRoundKey. 128-бітний стан шифрування s та 64 бітний твік T представляються у вигляді байтових матриць 4×4, де у матриці, яка представляє твік, два перших рядки займає власне твік, а інші рядки заповнені нулями. Значення $T = T_0 \parallel T_1 \parallel \ldots \parallel T_7$ ксориться з двома верхніми рядками стану у кожному раунді шифрування. Тобто i-тий раунд алгоритму AES

відбувається так: спочатку XOR з subkey, потім XOR з твіком, а далі відбуваються три інші перетворення одного раунда AES. Детально схему можна побачити на рисунку 2.6.

2.4.1 Атака відновлення ключа

В атаці використовується звернення до оракула для зашифрування блоків відкритого тексту. Атака побудована на інтегральній властивості AES. Блок шифротексту залежить від відповідного блоку BT та твіку. Твік складається трьох нульових біт, 32 біт nonce, які не змінюються та 29 біт, які показують номер блоку відкритого тексту. 29-біт представляють лічильник для побудови активного байта T7. Λ множина містить блоки відкритого тексту з номерами починаючи від 256 до 511. На цих номерах байт T7 приймає всі можливі значення. Завдяки зміні байта T7, підбираються потрібні блоки відкритого тексту. Це дає можливість додати один раунд, після якого стан шифру буде таким, щоб застосувати інтегральну властивість AES. Другий раунд збігається з першим раундом атаки на AES-128, але в кінець раунду додають активний твік байт. Твік, який додали в кінець 2 раунду привів до збалансованого байта у позиції 0. На третьому раунді маємо 1 збалансований байт перед SubBytes, саме цей байт стає невідомим. Після операції MixColumn перший стовпчик з невідомими байтами, а три інших з повністю активними. Щодо четвертого раунду, то виконується інтегральна властивість описана у пункті 2.2. Атаку можна продовжити на 7 раундів, складність, якої 2^{82}.

2.5 Аналіз схеми OCB

Шифр з автентифікацією OCB [13] (Offset Codebook) — один з переможців конкурсу CAESAR, розроблений Роговеєм (Rogaway. P) та Кровцем (Krovetz T.). Створений ними шифр є високопродуктивним,

тобто він є ефективними у застосуванні на платформах з високою продуктивністю таких як сервери, комп'ютери та смартфони. Раніше авторами була створена низка режимів роботи шифрів з автентифікацією OCB1, OCB2, OCB3. У 2001 році була розроблена перша версія даного режиму OCB1, яка забезпечувала конфіденційність та автентичність шифрування та стала стандартом для захисту бездротової мережі. Але OCB1 не розв'язує проблему автентифікації з асоційованими даними. Тому Роговей розробив OCB2, який покращує версію OCB1 та забезпечує автентифіковане шифрування з асоційованими даними. У 2019 році була опублікована стаття, в якій показано, що режим OCB2 нестійкий. У шифрі з автентифікацією OCB, який брав участь у конкурсі CAESAR, використовується більш надійний, швидкий, дешевий та ефективний режим OCB3 [14].

OCB [14]- це схема автентифікованого шифрування, яка містить налаштований блоковий шифр як внутрішній криптографічний примітив. BT шифрується за допомогою AES у режимі ECB з маскуванням кожного блоку до шифрування і після шифрування специфічним значенням Δ. В основі даного шифру лежить конструкція XEX (XOR-Encrypt-XOR), де шифрування відбувається так: $C = \Delta \oplus E_k(M \oplus \Delta)$, де Δ - певне зміщення. Шифр може підтримувати: ключі довжиною 128, 192 та 256 біт; блок відкритого тексту 128 біт, блок шифротексту 128 біт та тег довжиною 64, 96 та 128 біт.

Опишемо детально процес автентифікованого шифрування OCB з асоційованими даними на основі конструкції XEX. Спочатку генерується початкове зміщення $\Delta = H_k(N)$ отримане з nonce. Для генерації Δ була розроблена універсальна геш функція 'strech-then-shift'. При застосуванні її не викликається базовий блоковий шифр, що призводить до більшої ефективності. Потім, з використанням початкового зміщення за допомогою функції $\mathrm{Inc}(\Delta)$, будується послідовність зміщень Δ, кількість яких дорівнює кількості блоків для обробки відкритого тексту, асоційованих даних та генерації тегу. У залежності від того, чи блок є

повним чи неповним Inc функція обчислюється по-різному. Таке різноманіття зміщень необхідне для того, щоб шифр кожного блоку був унікальним і при цьому зберігалась можливість розпаралелювання шифрування блоків. Для повних блоків повідомлення функція збільшення $Inc_i(\Delta)$ складається з суми (XOR) початкового зміщення та $L(ntz(i))$, де $L(ntz(i))$ - функція обчислення кодів Грея від кількості останніх нулів у двійковому записі числа i $(ntz(i))$, де i - номер певного блоку. Позначимо $ntz(i) = j$, тоді $L(j) = a^{2+j}$, де a - корінь поліному генератора у полі GF(2^{128}). У такому випадку $a = 0^{126}10$. Для неповних, тобто тих, у яких присутній паддінг: $Inc_!(\Delta) = \Delta \oplus E_k(0^{128})$. Для генерації тегу: $Inc_*(\Delta) = \Delta \oplus a \cdot E_k(0^{128})$. Де Δ - початкове зміщення, $\cdot$ - операція множення у полі GF(2^{128}).

Обробка асоційованих даних. Асоційовані дані розбиваються на блоки 128-біт: $A \leftarrow A_1 \parallel A_2 \parallel \cdots \parallel A_{n-1} \parallel A_n^*$, де A_n^* - неповний блок, який вирівнюють додатковими бітами (10^*). Маємо початкове зміщення $\Delta \leftarrow 0^{128}$ та суму, яку отримаємо після обробки АД $Auth \leftarrow 0^{128}$. Для повних блоків обчислюється зміщення Δ за допомогою $Inc_i(\Delta)$, для неповних - $Inc_*(\Delta)$. Блоки асоційованих даних ксоряться з певним зміщенням та шифруються, загальна автентифікація: $Auth \leftarrow Auth \oplus E_k(A_i \oplus \Delta)$.

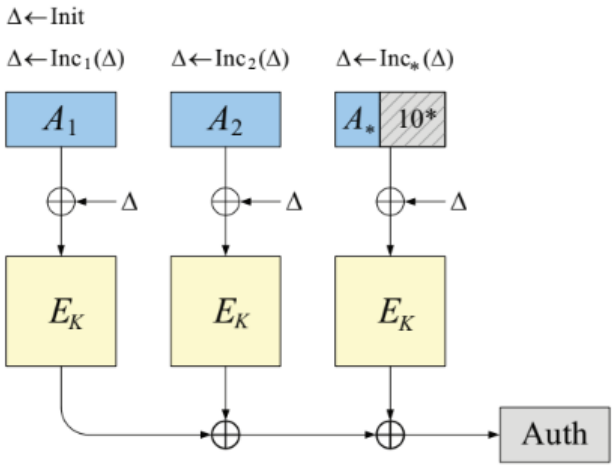

Рисунок 2.7 – Обробка АД в шифрі OCB

Шифрування. Відкритий текст розбивається на блоки 128-біт: $M \leftarrow M_1 \parallel M_2 \parallel \cdots \parallel M_{n-1} \parallel M_n^*$, де M_n^* - неповний блок, який вирівнюють додатковими бітами ((10^*). На кожній ітерації відбувається (XOR) відповідного зміщення з блоком відкритого тексту, потім AES шифрування. Результат шифрування сумується (XOR) з тим самим зміщенням, що і був застосований перед шифруванням, таким чином отримуємо блоки шифрованого тексту. Якщо останній блок неповний, то обчислюється $Pad \leftarrow E_k(\Delta)$. Шифрований блок - це сума неповного блоку та Pad.

Обчислення тегу. Для генерації тегу використовується контрольна сума (checksum) - це сума (XOR) всіх блоків відкритого тексту. Крім того, значення Checksum не залежить від того чи фінальний блок повний чи неповний. Контрольна сума сумується з відповідним зміщенням та відбувається шифрування. Результат шифрування сумується з автентифікованими асоційованими даними та отримується MAC-код.

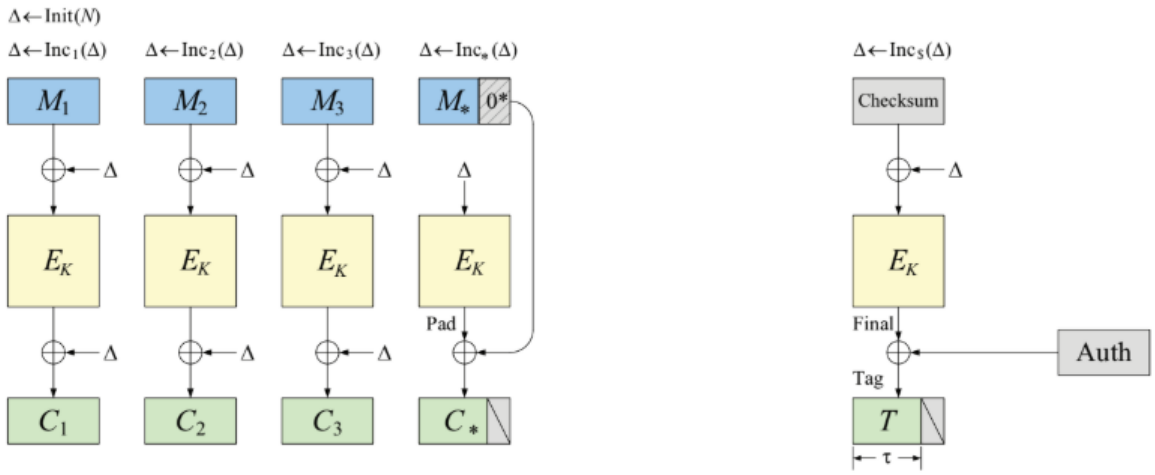

Рисунок 2.8 – Шифрування та обчислення тегу в OCB

Отже, при шифруванні кожен блок відкритого тексту шифрується незалежно від інших блоків з використанням твіку, який показує розміщення блока у відкритому тексті. Усі зміщення обчислюються заздалегідь, що підвищує ефективність. Завдяки тому, що до різних

блоків BT застосовуються різні зміщення, можна бути впевненим, що отримаємо різні ШТ, навіть коли блоки BT співпадають. У наслідок цього, у результаті шифрування отримуємо, що на різних позиціях відповідні шифровані блоки різні. Тобто зашифрований текст є випадковим. OCB має найменшу кількість викликів внутрішнього примітиву серед учасників.

2.6 Порівняння алгоритмів Silver, Kiasu, OCB

Ці три учасники конкурсу CAESAR є алгоритмами автентифікованого шифрування з налаштованим AES-128 в якості внутрішнього криптографічного примітива. Шифрування відкритого тексту та AD (принаймні всіх блоків, крім останнього, якщо він неповний) в усіх трьох алгоритмах відбувається у режимі ECB.

Різниця у будові цих шифрів в основному полягає у способі генерування та введення твіку і способі шифрування останнього неповного блоку, якщо такий є. Дещо розрізняються також алгоритми обчислення тегу. До схем Silver та Kiasu були застосовані атаки на основі інтегрального криптоаналізу, атак на OCB не було знайдено, хоча на попередні версії OCB1 та OCB2 вони були. Спосіб вводу твіку у схемі OCB — XEX, стійкість якого до атак з відкритим текстом доведена [15]. У алгоритмах Silver та Kiasu твік вводиться безпосередньо у ключ. Ця властивість і є однією з причин низької стійкості цих схем. Якщо твік вводиться таким чином, алгоритм шифрування кожного блоку є звичайним шифром AES, що створює можливість побудови криптоатаки на основі інтегрального аналізу.

Крім того, у цих трьох алгоритмах, різні способи шифрування останнього неповного блоку. У шифрі Silver цей алгоритм є настільки невдалим, що дозволив побудувати атаку відновлення останнього блоку за допомогою одного звернення до оракула (щоправда, у моделі nonce-repeating). Хоча в алгоритмі Silver твік вводиться досить складним

чином, проте він вводиться рідко, через 4 раунди, що якраз дозволяє використати інтегральну властивість AES для атаки. У Kiasu твік, навпаки, надто простий, вводиться звичайним XOR з ключем і має лише половину ненульових бітів, решта нулі. Це також слабкість, яка дозволяє реалізувати інтегральну атаку. Якщо ж подивитись на складність інтегральних атак, застосованих до Silver та Kiasu, важко визначити, який з цих шифрів більш стійкий. Немає криптоатак, які були б проведені при однакових умовах (nonce-respecting або nonce-repeating, однакова кількість раундів AES тощо). Наприклад, єдина інтегральна атака на Kiasu$\neq$ з відновленням ключа у моделі nonce-respecting при 7-цикловому AES має складність 2^{82} зашифрувань [16], в той час як атака відновлення ключа на 8-цикловий Silver має складність 2^{111} у nonce-repeating моделі. Проте існує атака підробки BT на 11-цикловий Silver у nonce-respecting моделі зі складністю приблизно 2^{50}. Можна з упевненістю тільки сказати, що на Silver було здійснено більше різних атак, заснованих на інтегральному аналізі, ніж на Kiasu.

Насамкінець зазначимо, що OCB є не тільки надійним, але і швидким алгоритмом, і серед причин цього можна назвати відсутність шифрування nonce при створенні твіку (мінус один виклик блокового шифру) та швидку рекурентну процедуру обчислення зсувів, які і є твіком.

Висновки до розділу 2

У даному розділі розглянутий метод інтегрального криптоаналізу і його застосування до блокових шифрів типу 'Квадрат', зокрема до AES. Детально описані схеми алгоритмів учасників конкурсу CAESAR Silver, Kiasu$\neq$, OCB та атаки на них, основані на інтегральному криптоаналізі. З проведеного аналізу можна зробити наступні висновки:

1) уведення твіку безпосередньо в ключ в схемах на налаштованому AES є слабкістю, яка допомагає будувати атаки, основані на інтегральному аналізі. Якщо ж вводити твік у ключ у таких схемах, то це треба робити

частіше, ніж 4 раунди;

2) алгоритм шифрування останнього неповного блоку має важливе значення і повинен бути ретельно обгрунтованим;

3) порівняти алгоритми Silver та Kiasu≠ з точки зору їх стійкості до інтегрального криптоаналізу важко, тому що до них застосовуються різні моделі атак. На Silver була розроблена ціла низка атак зі складністю від 1 до 2^{111}, у той час до Kiasu≠ була застосована тільки одна інтегральна атака знаходження ключа зі складністю 2^{82};

4) на даний час схема уведення твіку XEX дозволяє будувати ефективні та надійні схеми AEAD (До речі, інший переможець конкурсу CEASAR DEOXIS також побудований на схемі XEX).

ВИСНОВКИ

Автентифіковане шифрування — одна з найсучасніших та найбільш популярних галузей сучасної криптографії. Свідченням цього є всесвітній конкурс алгоритмів автентифікованого шифрування CAESAR, який закінчився близько року тому.

У ході дослідження було опрацьовано широкий спектр літератури, присвяченої як самому конкурсу та його учасникам, так і загальним питанням побудови схем автентифікованого шифрування та налаштованих блокових шифрах.

Були визначені учасники конкурсу CAESAR, які побудовані на налаштованих блокових шифрах і детально розглянуті троє з них: Silver, Kiasu та OCB. Проаналізовано особливості їх будови, способи налаштування (уведення твіку у внутрішній блоковий шифр).

Детально розглянутий метод інтегрального криптоаналізу і його застосування до алгоритмів автентифікованого шифрування, побудованих на налаштованих блокових шифрах, зокрема на налаштованому AES. Визначені вади алгоритмів Silver та Kiasu , які дозволили застосувати до них інтегральний криптоаналіз. Проведено порівняльний аналіз розглянутих алгоритмів Silver, Kiasu та OCB. З проведеного аналізу можна зробити наступні висновки:

1) уведення твіку безпосередньо в ключ в схемах на налаштованому AES є слабкістю, яка допомогає будувати атаки, основані на інтегральному аналізі. Якщо ж вводити твік у ключ у таких схемах, то це треба робити частіше, ніж через 4 раунди;

2) алгоритм шифрування останнього неповного блоку має важливе значення і повинен бути ретельно обгрунтованим;

3) порівняти алгоритми Silver та Kiasu$\neq$ з точки зору їх стійкості до інтегрального криптоаналізу важко, тому що до них застосовувались різні моделі атак. На Silver була розроблена ціла низка атак зі складністю від

1 до 2^{111}, у той час до Kiasu≠ була застосована тільки одна інтегральна атака знаходження ключа зі складністю 2^{82};

4) на даний час схема уведення твіку, застосована до OCB — XEX, дозволяє будувати ефективні та надійні схеми AEAD (До речі, інший переможець конкурсу CEASAR DEOXIS також побудований на схемі XEX).

Подальший розвиток даного дослідження можливий у плані розгляду інших алгоритмів автентифікованого шифрування на основі налаштованих блокових шифрів, їх класифікації з точки зору будови та стійкості, виявлення особливостей налаштування внутрішнього блокового шифру, що сприяють побудові ефективних криптоатак на них.

ПЕРЕЛІК ЛІТЕРАТУРИ

[1] Mihir Bellare and Chanathip Namprempre. “Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm”. In: *Journal of Cryptology* (2008).

[2] Phillip Rogaway. “Authenticated-encryption with associated-data”. In: Jan. 2002, p. 98. DOI: 10.1145/586123.586125.

[3] Cryptographic competitions. *CAESAR submissions*. http://competitions.cr.yp.to/caesar-submissions.html.

[4] Farzaneh Abed, Christian Forler, and Stefan Lucks. “General classification of the authenticated encryption schemes for the CAESAR competition”. In: *Computer Science Review* 22 (Oct. 2016). DOI: 10.1016/j.cosrev.2016.07.002.

[5] Liskov Moses, Rivest Ronald L Wagner, and David. “Tweakable Block Ciphers”. In: *Journal of Cryptology* (2011).

[6] Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. *Tweaks and Keys for Block Ciphers: the TWEAKEY Framework*. Cryptology ePrint Archive, Report 2014/831. https://eprint.iacr.org/2014/831. 2014.

[7] Joan Daemen, Lars Knudsen, and Vincent Rijmen. “The block cipher Square”. In: *Lecture Notes in Computer Science* 1267 (Oct. 1998). DOI: 10.1007/BFb0052343.

[8] Lars Knudsen and David Wagner. “Integral Cryptanalysis”. In: vol. 2365. Feb. 2002, pp. 112–127. DOI: 10.1007/3-540-45661-9_9.

[9] Daniel Penazzi and Miguel Montesg. *Silver*. http://competitions.cr.yp.to/caesar-submissions.html. 2014.

[10] Jérémy Jean, Yu Sasaki, and Lei Wang. "Analysis of the CAESAR Candidate Silver". In: vol. 9566. Jan. 2016, pp. 493–509. ISBN: 978-3-319-31300-9. DOI: 10.1007/978-3-319-31301-6_28.

[11] Jérémy Jean and Ivica Nikolić andThomas Peyrin. *KIASU*. http://competitions.cr.yp.to/caesar-submissions.html. 2014.

[12] Ted Krovetz and Phillip Rogaway. "The Software Performance of Authenticated-Encryption Modes". In: May 2011, pp. 306–327. DOI: 10.1007/978-3-642-21702-9_18.

[13] Phillip Rogaway et al. *OCB Mode*. Cryptology ePrint Archive, Report 2001/026. https://eprint.iacr.org/2001/026. 2001.

[14] Ted Krovetz and Phillip Rogaway. "The Software Performance of Authenticated-Encryption Modes". In: May 2011, pp. 306–327. DOI: 10.1007/978-3-642-21702-9_18.

[15] Phillip Rogaway. "Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC". In: Dec. 2004, pp. 16–31. DOI: 10.1007/978-3-540-30539-2_2.

[16] Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. *Square Attack on 7-Round Kiasu-BC*. Cryptology ePrint Archive, Report 2016/326. https://eprint.iacr.org/2016/326. 2016.